DOCUMENT RESUME

ED 252 557 TM 850 026

AUTHOR McArthur, David
TITLE Diagnostic Testing Package DX v 2.0 Technical Specification. Methodology Project.
INSTITUTION California Univ., Los Angeles. Center for the Study of Evaluation.
SPONS AGENCY National Inst. of Education (ED), Washington, DC.
PUB DATE Nov 84
GRANT NIE-G-84-0112-P1
NOTE 18p.; For related document, see TM 850 027. Computer programs included in this document were written by David McArthur and C. R. Morris.
PUB TYPE Computer Programs (101) -- Reports - Descriptive (141)

EDRS PRICE MF01/PC01 Plus Postage.
DESCRIPTORS Adaptive Testing; *Computer Assisted Testing; *Computer Software; *Diagnostic Tests; Elementary Secondary Education; Flow Charts; Higher Education; *Test Construction
IDENTIFIERS *DX Diagnostic Testing System; *PASCAL Programing Language

ABSTRACT

This paper contains the technical specifications, schematic diagrams, and program printout for a computer software package for the development and administration of diagnostic tests. The second version of the Diagnostic Testing Package DX consists of a PASCAL-based set of modules located in two main programs: (1) EDITTEST creates, modifies, and reviews the test materials which are to appear on screen; and (2) RUNTEST administers, manages, and scores the real-time testing session. EDITTEST creates two files for each test: (1) "filename" Q.TEXT, a test file containing registers for each test item; and (2) "filename," a datafile containing the test text. RUNTEST opens the EDITTEST files for input and an output file, "studentname." TEXT, which summarizes a student's test performance. This last file follows the format requirements of STATPAK, a microcomputer statistical package which is being adapted for a third main program for the DX package. (BS)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

ED252557

U.S. DEPARTMENT OF EDUCATION
NATIONAL INSTITUTE OF EDUCATION
EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)
☑ This document has been reproduced as received from the person or organization originating it.
☐ Minor changes have been made to improve reproduction quality.
• Points of view or opinions stated in this document do not necessarily represent official NIE position or policy.

DELIVERABLE - NOVEMBER 1984

METHODOLOGY PROJECT

Diagnostic Testing Package
DX v 2.0 Technical Specifications

DAVID McARTHUR
Project Director

Grant No.
NIE-G-84-0112, P1

Center for the Study of Evaluation
Graduate School of Education
University of California, Los Angeles

TM 850 026

The project presented or reported herein was performed pursuant to a grant from the National Institute of Education, Department of Education. However, the opinions expressed herein do not necessarily reflect the position or policy of the National Institute of Education, and no official endorsement by the National Institute of Education should be inferred.

Table of Contents

	Page
Introduction	1
Schematic diagrams	3
Program listings	5

Diagnostic Testing Package DX v 2.0 Technical Specifications

David McArthur

Introduction

The Diagnostic Testing Package DX, in its second version, consists of a PASCAL-based set of modules located in two main programs. The first, EDITTEST, is a packet for creating, modifying, and reviewing test materials which are to appear on screen. The second, RUNTEST, is a packet which administers, manages, and scores a real-time testing session using the above test materials.

PASCAL was adopted as the language of choice for the DX package for three separate reasons. First, the language is a central part of the p-system environment, which generates compact code readily transferable between competing brands of microcomputer. Second, PASCAL supports random-access file manipulation and certain string-oriented algorithms which are relied upon for rapid presentation of materials and processing of responses, without mandating that the user purchase specialized equipment. Third, because PASCAL is a highly structured language, further program development is easier to consider (both in terms of expansion of capabilities and optimization of code) than in those programming languages with less structure or readability.

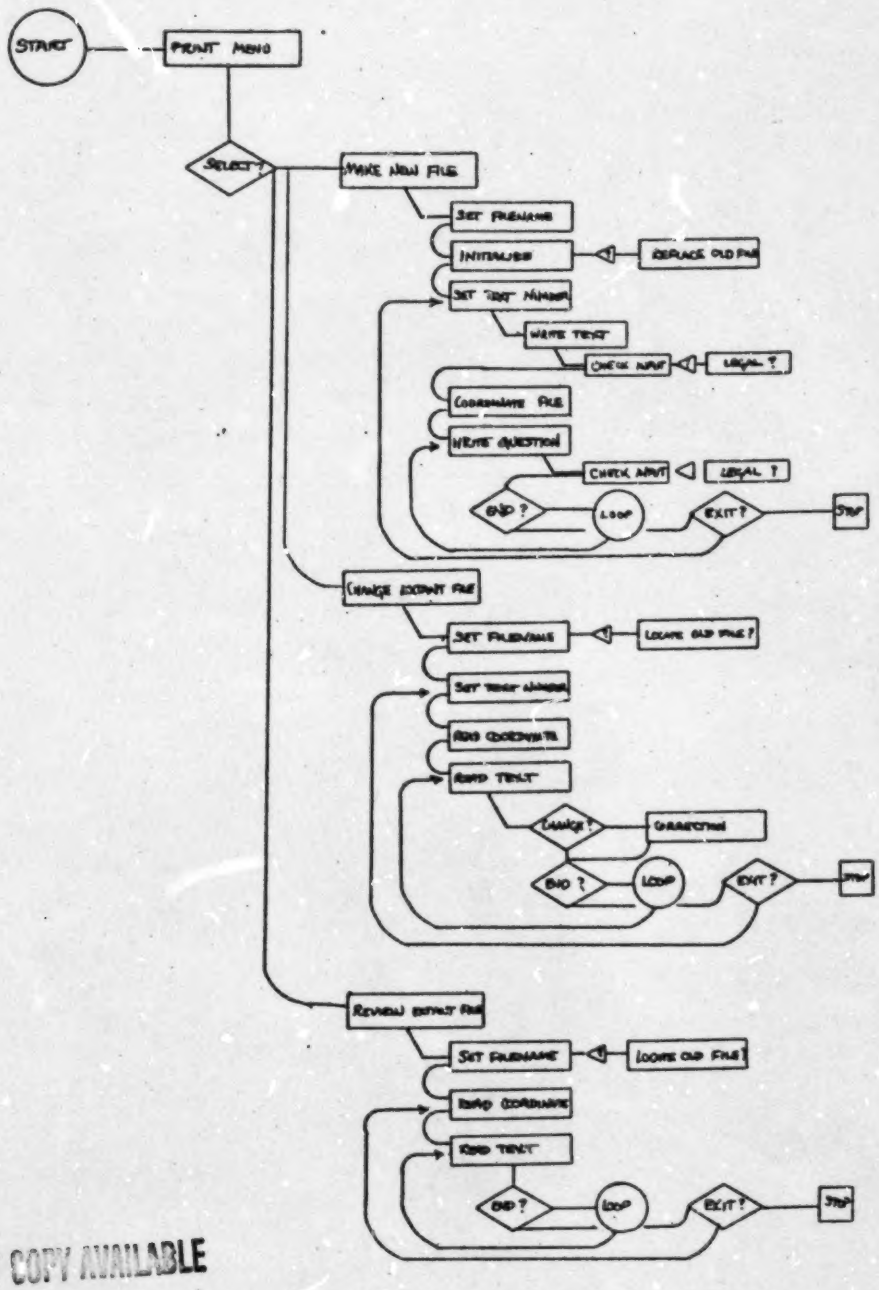
EDITTEST creates two files for each separate test to be administered. The first is filename.Q.TEXT, a text file containing registers for each separate test item. This file is accessed by RUNTEST to determine the size of the test, the number of items that appear with each separate test question, and the location of these materials in the second file. The second file is filename, a datafile containing the text of the test.

Because of the random-access requirements, this file is not ordinary ASCII code and may not be edited directly. EDITTEST therefore contains modules which allow the text within this file to be read to screen, and within certain limits to be modified a line at a time. The asterisk symbol (*) denotes the end of a test question, or the end of a test item followed by the correct answer for that item.

RUNTEST opens the datafile filename, and the textfile filename.Q.TEXT for input, and an output file studentname.TEXT which at conclusion contains a summary of the student's performance on the test. This last file follows the format requiremetns of UCLA's STATPAK, a general-purpose microcomputer statistical package, which is being adapted and optimized for use as a third main program for the DX package. It will allow the instructor to rapidly acquire summary data for a series of students who have taken the same test. studentname.TEXT will also contain the program's narrative summary of individual performance, which may be seen on screen by the instructor at the close of testing.

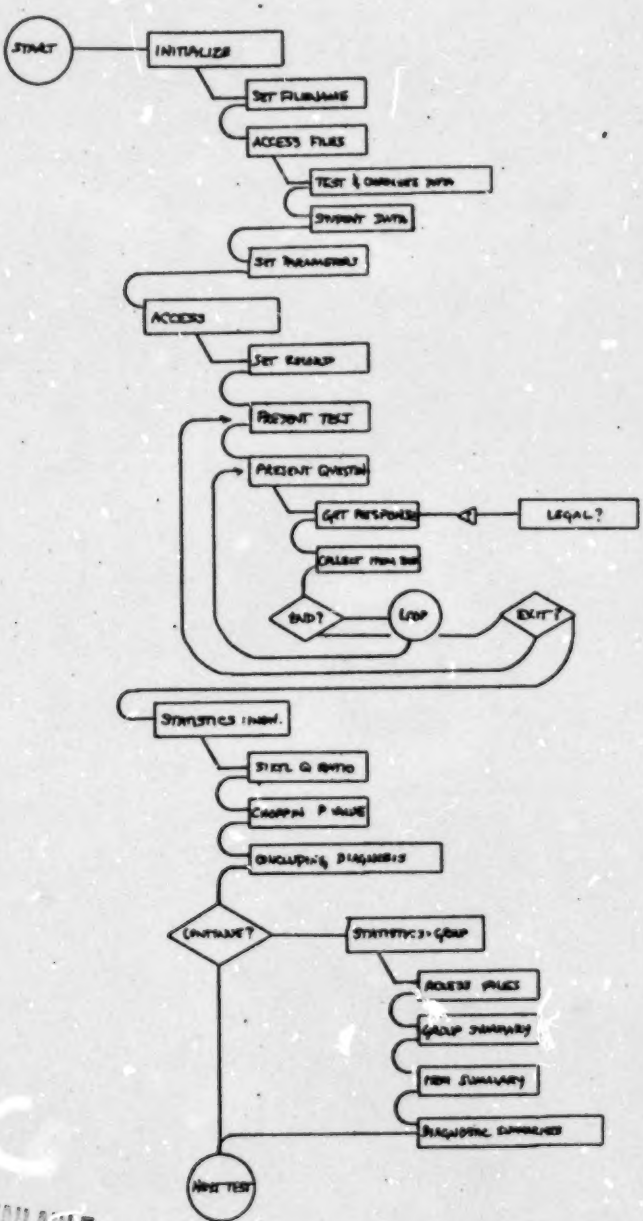
The following schematic diagrams show program operations for EDITTEST and RUNTEST.

DX | Edittest



BEST COPY AVAILABLE

DX | Runtest



BEST COPY AVAILABLE

```
program EDITTEST;   **DRAFT**                    (* edition 25-nov-84 *)
{ .......a program to create, review, and modify a
          microcomputer-managed diagnostic testing session.........
                                   <^>
   Developed at UCLA under the auspices of The National Institute of Education,
          Grant No. G-84-0112, PI by D.L.McArthur and C.R.Morris.......)
                                   <^>
  CONST
        Maxlines  = 20;
        Maxitem   = 25;
        Disc      = 'DAT:';
        dash='--------------------';
  TYPE
    item = record
        material  :array[1..Maxlines] of string[80];
        end;
  CS  = set of char;
  VAR   iocheck  :integer;
        count, i, j,questot   :integer;
        recno, itemn, questn, leveln, quesloc  :integer;
        keycode:    char;  U   :string[3];
        code,check,place     :string[1];
        temporary,filename, phrase,trk    :string[80];
        Questions    :file of item;
        OUT    :text;

procedure INITIALIZE;
      begin
      with Questions^ do count:= 0;questot :=0;i:=0;
         recno:=0;itemn:=0 ;questn:=0;leveln:=5;quesloc:=Maxitem;
         (*$I-*)  reset (OUT,trk);
                  iocheck := ioresult;
         (*$I+*)
      end;

procedure INSERTTEXT (i :integer);
      begin;
      with Questions^ do begin;
      writeln('  Insert line # ',i,' of the text here,   or  <*>');
            readln (material[i]);
            phrase :=(material[i]);
            code :=copy(phrase,1,1);
            end;
      end;

procedure CHECKINPUT (var checknum  :integer);
      var input: char; output  :integer;
      begin
          readln (input);
          if input = 'stop' then begin
              writeln ('  Thanks.... we now close all files');
              close (questions, lock);
              close (OUT,lock);
                exit(program);  end;
          output:= ORD(input) - ORD('0');
          if (output >=0) and (output <=10) then begin
              output:= ord(input) - ord('0');
              checknum :=output;end
          else begin
              writeln ('...Sorry, that character isn''t legal here...');
              writeln ('     Type in the correct number:  ');
              readln (checknum)
```

BEST COPY AVAILABLE

```
                end;
    end;                                        (* end checkinput *)

function LOOKEND  (code: string):integer;
   var  found: boolean;  index  :integer;
   begin
         index :=1;  found := false;
         while not found and (index <= Maxlines) do begin
             if (code = '*') then begin
                  found := true;
                  lookend := 1
                  end
             else begin
                  index := index+1;
                  lookend := 0
                  end
             end;
   end;

procedure WRITEITEM;
    begin;
        quesloc := quesloc + 1;
          writeln(dash,dash);
          writeln('Enter * as the last line');
          writeln(dash,dash);
          writeln;
        seek (Questions, quesloc);
        with Questions^ do begin;
        repeat;
           i:=i+1;
           INSERTTEXT (i);
           until (LOOKEND(code)=1)
           end;
        seek (Questions,quesloc);
        put (Questions);
     end;

procedure SETQUESTNUM;
   var changenum: char;
   begin;
      if (questn>Maxitem) then exit(SETQUESTNUM);
      itemn :=itemn +1;
      WRITEITEM;
          writeln (' Enter the number of questions that go with this item: ');
          CHECKINPUT (questot);
          with Questions^ do begin
              count :=questot;
              end;
   end;

procedure SETFILENAME;
   begin
       write ('  Enter the name for this Test File :  ');
       readln (filename);
       if (filename = 'stop') or (filename = 'STOP') then begin
           writeln ('.....ending at your request.....');
           exit (program);
           end;
       filename := concat(DISC,filename);
       trk := concat(filename,'.0.TEXT');
   end;

procedure DIRECTORY;
   begin; {to be implemented: access directory of disc DAT:}
   end;
```

BEST COPY AVAILABLE

```
procedure SEERECNUM;
   begin;
       read (OUT,recno,itemn,questn,leveln,quesloc);
       writeln ('  file contains   ',
            recno,' ',itemn,' ',questn,' ',leveln,' ',quesloc);
       iocheck:=ioresult;
   end;

procedure PUTRECNUM;
     begin;
     writeln (OUT,recno:5,itemn:5,questn:5,leveln:5,quesloc:5);
     end;

procedure ACCESSMARE;
     var ch: char;
     begin;
     iocheck:=0;
        SETFILENAME;
        INITIALIZE;
        (*$I-*) reset (Questions,filename);
        if ioresult>0 then iocheck :=1;
        (*$I+*)
        close (Questions);
        close (OUT);
        if iocheck=0 then begin
             writeln ('   File by that name already exists.....');
             DIRECTORY;
             write   ('  Would you like to replace it?  <Y><N> ');
             readln (ch);
             if (ch = 'y') or (ch = 'Y') then begin;
                 rewrite (Questions,filename);
                 rewrite (OUT,trk);
                 end
             else begin
                 SETFILENAME;
                 rewrite (Questions, filename);
                 rewrite (OUT,trk)
                 end;
         end
    else begin;
         rewrite (Questions, filename);
         rewrite (OUT,trk);
         end;
    repeat;
    recno := recno+1;
         questn :=questn+1;
         if (questn>questot) then begin;
                 SETQUESTNUM;
                 questn  :=1;
                 end;
         PUTRECNUM;
         seek (Questions, recno);
         with Questions^ do begin;
         i:=0;
             if (recno=0) then begin
                              writeln(dash,dash);
                              writeln('Enter * as the last line');
                              writeln(dash,dash);writeln
                              end;
             repeat;
                     i:=i+1;
                     INSERTTEXT (i);
             until (LOOKEND(code)=1)
             end;
          seek (Questions,recno);
          put (Questions);
```

BEST COPY AVAILABLE

```
            close(Questions,lock);writeln( 'We have now closed files ');
            close(OUT,lock)
    end;  (*.....................end accessmake *)

procedure ACCESSCHANGE;
  var changenum: char;
  begin
    SETFILENAME;
    (*$I-*) reset (Questions,filename);
            iocheck:=ioresult;
            reset (OUT,trk);
    (*$I+*)
    if iocheck<>0 then begin
       writeln ('File not found'); DIRECTORY;SETFILENAME;
       reset (Questions,filename) end;
    repeat;
      SEERECNUM;
      if (recno<0) then exit (ACCESSCHANGE);
      seek(Questions,recno);
      get (Questions);
      with Questions^ do begin
        i:=0;
        if (recno =0) then begin
          writeln('  Number of questions is: ',count);
          write  ('  Enter correct number of questions: ');
          CHECKINPUT (count);
          questot:=count
          end;
        repeat;
          i:=i+1;
          phrase:=(material[i]);
          check:=copy(phrase,1,1);
          writeln(material[i]);
          writeln('  Enter corrections to this line...');
          readln (material[i]);
          temporary:=(material[i]);
          place :=copy(temporary,1,1);
          if (length(temporary)=0) then material[i]:=phrase;
          until (LOOKEND(check)=1)or(LOOKEND(place)=1);
          if (LOOKEND(place)=1) and (length(temporary)>1) then begin
            repeat;
              i:=i+1;
              writeln('  Enter new line of text: ');readln(material[i]);
              phrase:=(material[i]);code:=copy(phrase,1,1);
              until (LOOKEND(code)=1);end;
            end;
      seek (Questions,recno);
      put (Questions);
      until (recno>=questot);
      close (Questions,lock);
      close (OUT,lock)
    end;

procedure ACCESSREAD;
  var resp: string[1];
  begin;
    SETFILENAME;
    (*$I-*) reset (Questions, filename);
            iocheck := ioresult;
    (*$I+*)
    if iocheck<>0 then begin writeln ('File not found  ...');
            DIRECTORY;SETFILENAME;
            reset (Questions,filename);
            end;
    with Questions^ do
      questn := count;
```

BEST COPY AVAILABLE

BEST COPY AVAILABLE

```
      repeat;
          SEERECNUM;
          writeln(dash);
          seek(Questions,recno);
          get (Questions);
          with Questions ^ do begin
            i:=0; repeat;
                  i:=i+1;
                  writeln(material[i]);
                  phrase:=(material[i]);
                  code:= copy(phrase,1,1);
                  until (LOOKEND(code)=1)
          end
      until (recno>= questn);
      write ('   Press <RETURN> key when you have finished reading...');
      readln (resp);
      if (length(resp)=0) then begin;
      close(Questions,lock);close(OUT,lock);end
  end;

procedure  SHOWMENU;
  begin writeln (dash,dash,dash);
        writeln ('Enter the letter for the procedure you wish to use:');
        writeln;
        writeln ('  W -- write new test items ');
        writeln ('  C -- revise existing test items ');
        writeln ('  R -- Review the contents of the test');
        writeln ('  Q                                -- quit');
        writeln;
        writeln (dash,dash,dash,dash);
        write ('..... Your choice is ?  ');
  end;

procedure PRINTMENU;
  var checkcode: char; checker,entry: CS;
  begin; SHOWMENU;
        readln (checkcode);
        checker:=[checkcode];
        entry :=['W','C','R','Q','w','c','r','q'];
        if checker<= entry then keycode := checkcode
        else begin writeln ('...Sorry, incorrect code');
                   write   ('Enter the correct code : ');
                   readln (keycode)
             end;
  writeln(dash,dash,dash,dash);
  end;


BEGIN (*.....................................................MAIN.....*)
      repeat; PRINTMENU;
  writeln(dash,dash,dash,dash);
      case keycode of
        'W','w': ACCESSMAKE;
        'C','c': ACCESSCHANGE;
        'R','r': ACCESSREAD;
        end;
      until (keycode = 'q') or (keycode = 'Q');
      writeln ('   Thanks....that is the end of the program');
  writeln(dash,dash,dash,dash);
END.
```

BEST COPY AVAILABLE

```
Program RUNTEST;          **DRAFT**          (*  edition 22-Nov-84 *)
   { .......a program to administer and score a
            microcomputer-managed diagnostic testing session........
                                        <^>
   Developed at UCLA under the auspices of The National Institute of Education,
            Grant No. G-84-0112, P1 by D.L.McArthur and C.R.Morris.......)
                                        <^>
   USES TRANSCEND;
   CONST
        Maxinfo = 4;
        Maxquest = 25;
        Maxline = 20;
        Maxitem = 25;
        Disc = 'DAT:';
        P1 = 0.5; P2 = 0.75;  P3 = 0.95;
        Maxhead = 9;
        Screensize= 23;
   TYPE
        item = record
             material  :array[1..Maxline] of string[80]
             end;
        testinfo = array[1..Maxitem] of record
             testresp :array [1..Maxquest] of integer;
             testinfo :array[1..Maxinfo,1..Maxquest] of integer;
             testhypo :array[1..Maxinfo,1..Maxquest] of integer;
        end;
   VAR
        iteminfo :testinfo;
        H1,H2,H3,H4, NumofItem,Attempt,Correct  :integer;
        recno, itemn, questn, leveln, quesloc  :integer;
        testnum,questnum,i,j,m,count,place   :integer;
        newline,filename,phrase,trk  :string[80];
        keyword  :string[10];
        TH1,TH2,TH3,TH4, coder, Response, TempResp  :string[1];
        keycode, code  :char;
        Questions  :file of item;
        OUT :text;

procedure INITIALIZE;
   var i,j,k,l :integer;
   begin
        for i:= 1 to Maxitem do
          with iteminfo[i] do begin
            for j:= 1 to Maxinfo do
              for k:= 1 to Maxquest do begin
                testinfo[j,k] :=0;testhypo[j,k]:=0 end;
              for l:=1 to Maxquest do testresp[l]:=0
          end;
        questnum:=0
   end;

procedure SIXTL;
   var RPC,X,Y,N,K,i,j,Test,Ques,A,C,num:  integer;
       PC: real;
       keyword, heading, filename: string;
       datafile: text;
   procedure COMPUTE;
     var Q1,Q2,Q3,U,U1,U2,U3,U4,U5,V,V1,V2,V3,V4,V5: real;
     begin;
       Y :=N-X;
       U := (P1 * (K-1)) +1;
```

BEST COPY AVAILABLE

```
        U1 := EXP( X*LN(U) );
        V := (1 - P1);
        V1 := EXP( Y*LN(V) );
        Q1 := U1 * V1;
        U2 := (P2 * (K-1)) +1;
        U3 := EXP( X*LN(U2) );
        V2 := (1-P2);
        V3 := EXP( Y*LN(V2) );
        Q2 := U3 * V3;
        U4 := (P3 * (K-1)) +1;
        U5 := EXP( X*LN(U4) );
        V4 := (1-P3);
        V5 := EXP( Y*LN(V4) );
        Q3 := U5 * V5;
        writeln ('  Sixtl''s Q :  ',Q1:3:8, Q2:3:8, Q3:3:8);
      end;
      begin;
        X :=0; n :=0; K :=4;
        filename := concat(DISC,filename,'.STAT');
        reset (datafile, filename);
        begin;
           writeln;
           writeln(' Sixtl''s Q values for P = 0.5,  P=0.75,  P=0.95');
           writeln;
           end;
        while not eof(datafile) do begin
           for i := 1 to Maxhead do
             readln (datafile, heading);
           for j := 1 to Maxquest do begin
             read (datafile, Test, Ques);
             if (Test <>0) and Ques <>0) then begin
               read (datafile, A);
               if (A = 1) then begin
                 N := N+1;
                 readln (datafile, C);
                 if (C=1) then
                   X := X+1;
                 begin
                   write (' # of attempts = ',N);
                   writeln ('   # Correct = ',X);
                   COMPUTE
                 end
               end
           end;
        PC := 100 * (X/N);
        RPC := round (PC);
        writeln;
        write ('For ',N,' attempts with ',X,' correct ');
        writeln (' Percent = ',RPC);
        writeln;
        close (datafile,lock)
        end
    end;


function LOOKEND (code:string):integer;
    var foundit:boolean; index:integer;
    begin
      index:=1;foundit:=FALSE;
      while (not foundit) and (index <= Maxline) do begin
        if (code = '*') then begin
          foundit:=TRUE;LOOKEND:=1 end
        else begin
          index :=index+1; LOOKEND:=0 end
      end
  end;
```

BEST COPY AVAILABLE

```
procedure COLLECTITENDATA;
   var respnum: integer;
   begin
     if Response ='a' then respnum:=1;
     if Response ='b' then respnum:=2;
     if Response ='c' then respnum:=3;
     if Response ='d' then respnum:=4;
   with iteminfo[testnum] do begin
        testresp[recno]:=respnum;
        testinfo[1,recno]:=NumofItem;
        testinfo[2,recno]:=Attempt;
        testinfo[3,recno]:=Correct;
        testhypo[1,recno]:=H1;
        testhypo[2,recno]:=H2;
        testhypo[3,recno]:=H3;
        testhypo[4,recno]:=H4;
      end
    end;

procedure SAVEDATA;
   var i,j,count: integer;
       filename:string; outfile:interactive;
  procedure HEADER;
     begin;
       writeln(outfile,'Testnum?=0, ');
       writeln(outfile,'Questnum?=0, ');
       writeln(outfile,'Attempt[(1) yes (2) no]?=0, ');
       writeln(outfile,'Correct[(1) yes (2) no]?=0, ');
       writeln(outfile,'Response[(1) a (2) b (3) c (4) d]?=0 ');
       writeln(outfile,'HYP1[(1) yes (2) no]?=0, ');
       writeln(outfile,'HYP2[(1) yes (2) no]?=0, ');
       writeln(outfile,'HYP3[(3) yes (2) no]?=0, ');
       writeln(outfile,'HYP4[(1) yes (2) no]?=0, ');
     end;
     begin
       write (' Enter student name for output file :  ');
       readln (filename);
       if (pos('.TEXT',filename)=0) and (pos('.text',filename)=0) then
         filename := concat(filename,'.TEXT');
       filename := concat(Disc,filename);
       rewrite (outfile,filename);
       HEADER;
         for i:=1 to testnum do
           with iteminfo[i] do
           for j:=1 to Maxquest do begin
             write (outfile, i:5);
             write (outfile, testinfo[1,j]:5);
             write (outfile, testinfo[2,j]:5);
             write (outfile, testinfo[3,j]:5);
             write (outfile, testresp[j]:5);
             write (outfile, testhypo[1,j]:5);
             write (outfile, testhypo[2,j]:5);
             write (outfile, testhypo[3,j]:5);
             writeln (outfile,testhypo[4,j]:5);
             end;
        writeln (outfile, '    0    0    0    0    0    0    0    0   0* ');
       close (outfile, lock)
     end;

procedure SETFILENAME;
    begin; write ('  Enter name of file to be read : '); readln (filename);
           if (filename='stop') or (filename='STOP') then begin
              SAVEDATA; writeln ('  You have reached the end...');
              exit (program); end;
           filename := concat (Disc,filename);
```

BEST COPY AVAILABLE

```
        trk := concat (filename,'.Q.TEXT');
      end;

procedure SETRECNUM;
   begin;
   read (OUT, recno, itemn, questn, leveln, quesloc);
   end;

procedure PROCESSRESP;
   begin;  if Response= 'A' then response :='a';
           if Response= 'B' then Response :='b';
           if Response= 'C' then Response :='c';
           if Response= 'D' then Response :='d';
         NumofItem :=recno;
           if TH1='1' then h1:=1 else H1:=2;
           if TH2='1' then H2:=1 else H2:=2;
           if TH3='1' then H3:=1 else H3:=2;
           if TH4='1' then H4:=1 else H4:=2;
           if (length(Response)=0) then begin
             Attempt:=2;Correct:=0;end;
           if (length(Response)>0) then
              if (TempResp=Response) then begin
                 Correct:=1;Attempt:=1 end
              else begin
                 Correct:=2;Attempt:=1 end
         end;

procedure ACCESS;
   begin; while TRUE do begin
      SETFILENAME;
      reset (Questions, filename);
      reset (OUT,trk);
      repeat
          SETRECNUM;
          seek (Questions, quesloc);
          get (Questions);
          with Questions^ do begin
               GOTOXY(0,1);
               i:=1;
               count :=0;
               coder:=' ';
               while (LOOKEND(coder)<>1) do begin;
                 phrase:=(material[i]);
                 coder :=copy(phrase,1,1);
                 if (coder<>'*') then writeln (material[i]);
                 count:=i;
                 i:=i+1;
                 end;
            i:=1;
          seek (Questions, recno );
          get (Questions);
            place :=Screensize-count;
            coder:=' ';
            GOTOXY(0,count );
            while (LOOKEND(coder)<>1) do begin
              phrase := (material[i]);
              coder  := copy(phrase,1,1);
              if (coder <> '*') then writeln (material[i]);
              i:=i+1;
              end;
            write ('  Type your response  : ');readln(Response);
            i:=i-1;
            newline := (material[i]);
            TempResp := copy(newline,2,1);
            TH1:= copy(newline,3,1);
            TH2 :=copy(newline,4,1);
```

BEST COPY AVAILABLE

```
            TH3 :=copy(newline,5,1);
            TH4:= copy(newline,6,1);
            PROCESSRESP;
            COLLECTITEMDATA;
            end
        until (recno>=Maxquest );
        close (Questions,lock);
        close(OUT,lock)
        end
    end;

BEGIN (*.....................................main program.....*)
  INITIALIZE;
  ACCESS;
  SIXTL;
END.
```